ECE3120: Computer Systems
Chapter 4: Advanced Assembly Programming

Manjeera Jeedigunta
http://blogs.cae.tntech.edu/msjeedigun21
Email: msjeedigun21@tntech.edu
Tel: 931-372-6181, Prescott Hall 120

Objectives of this chapter

- Access & manipulate arrays, vectors and strings
- Access parameters stored in the stack and manipulate stack data structure
- Perform binary to ASCII string and ASCII string to binary conversion
- Write subroutines to perform certain functions
- Make subroutine calls

Introduction

Program = data structures + algorithm

Data structures: how information is organized to be processed efficiently;

Algorithm: systematic method to process information; adding/deleting elements, traversing/searching a data structure, sorting a data structure…

- Data structures to be discussed
  1. stacks: a last-in-first-out (LIFO) data structure
  2. arrays: a set of elements of the same type
  3. strings: a sequence of characters terminated by a special character

- Algorithms can be represented in procedural steps or flowcharts;

Stack

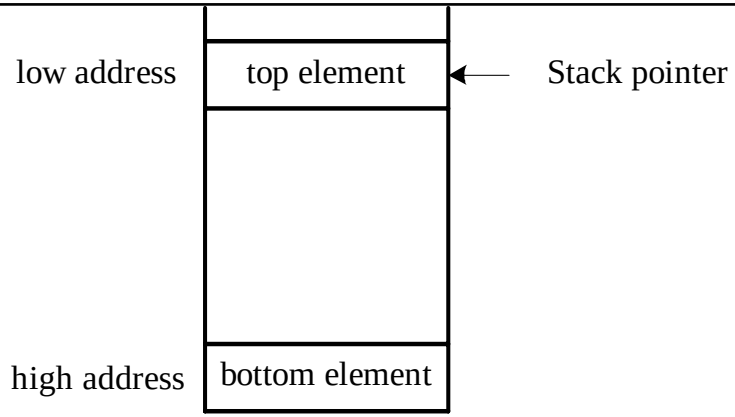

Figure 4.1 Diagram of the 68HC12 stack

Push: add a new item to the top;

Pop or Pull: remove an item from the top;

Stack overflow:

Stack underflow:

68HCS12 Support for the Stack Data Structure

- A 16-bit stack pointer (SP)

- Instructions and addressing mode

Table 4.1 68HC12 push and pull instructions and their equivalent load and store instructions

Mnemonic	Function	Equivalent instruction
psha	push A into the stack	staa 1, -SP
pshb	push B into the stack	stab 1, -SP
pshc	push CCR into the stack	none
pshd	push D into stack	std 2, -SP
pshx	push X into the stack	stx 2, -SP
pshy	push Y into the stack	sty 2, -SP
pula	pull A from the stack	ldaa 1, SP+
pulb	pull B from the stack	ldab 1, SP+
pulc	pull CCR from the stack	none
puld	pull D from the stack	ldd 2, SP+
pulx	pull X from the stack	ldx 2, SP+
puly	pull Y from the stack	ldy 2, SP+

Example 4.1: What would the contents of the stack after the execution of the instructions be?

```
lds     #$1500      ;initializes SP=$1500
ldaa    #$20        ;A←$20 (8-bit)
staa    1,-SP       ;Auto Pre-decrement Idx Addr
ldab    #40         ;B←40
staa    1,-SP       ;stores A in SP-1, then decr SP
ldx     #0          ;X←0 (16-bits)
stx     2,-SP       ;store 0 on top 2 bytes of stack
```

Example 4.1: What would the contents of the stack after the execution of the instructions be?

```
lds     #$1500      ; SP=$1500
ldaa    #$20        ;A←$20 (8-bit)
staa    1,-SP       ;SP=$14FF → contents=$20
ldab    #40         ;B←40
staa    1,-SP       ;SP=$14FE →contents=40
ldx     #0          ;X←0 (16-bits)
stx     2,-SP       ;SP=$14FC→contents=0
```

Indexable Data Structures (arrays)

- Vectors and matrices are indexable data structures.
- The first element of a vector is associated with the index 0 in order to facilitate the address calculation.
- Assemblers directives db, dc.b, fdb are used to define arrays of 8-bit elements.
- Assemblers directives dw, dc.w, and fdb are used to define arrays of 16-bit elements.
- directives ds,rmb, ds.b are used to reserve memory space for arrays with 8-bit element;
- directives ds.w and rmw are used to reserve memory space for arrays with 16-bit element;

Next…

- Indexable Data Structures
- Binary search